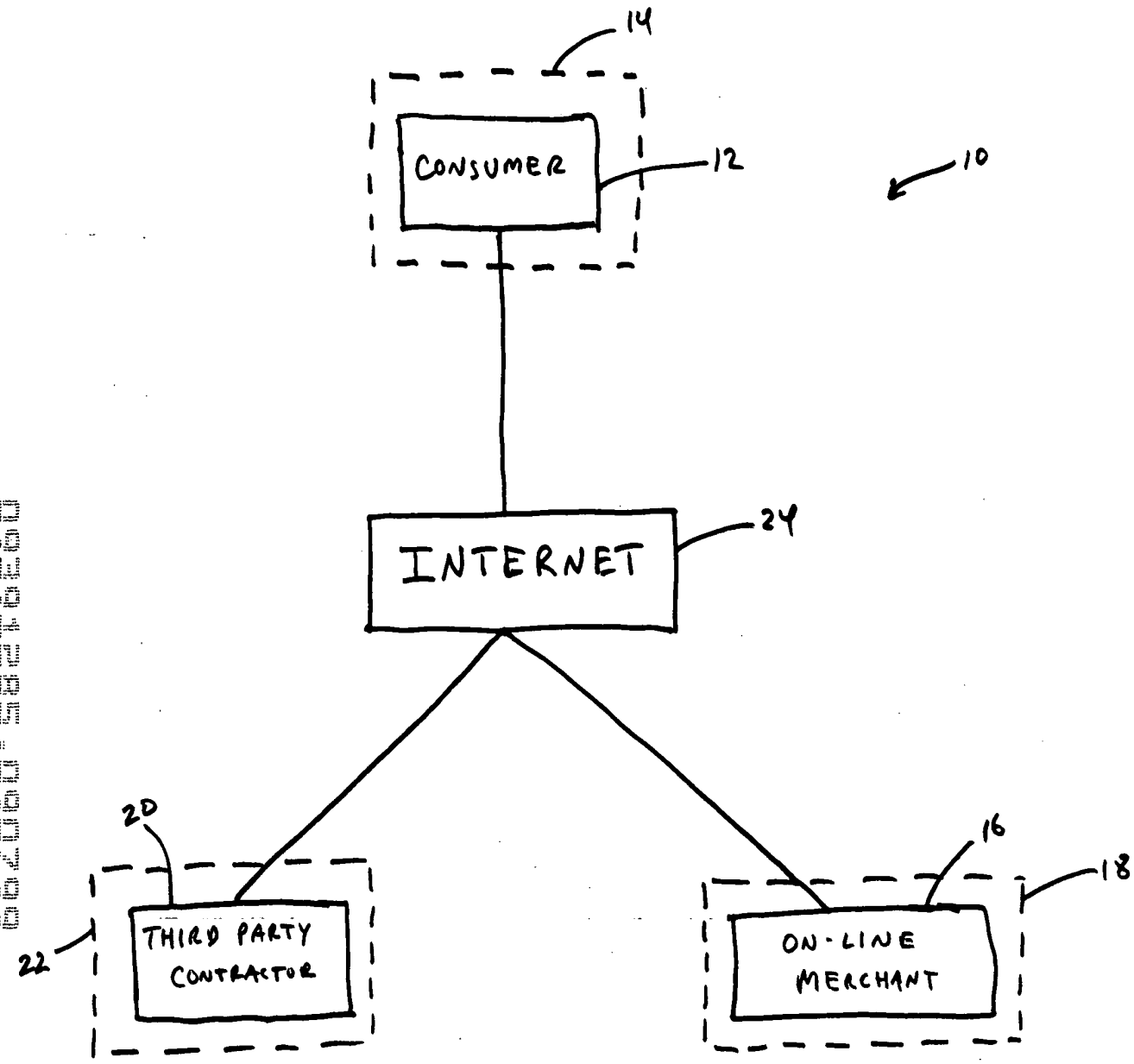
14
CONSUMER
12
10
INTERNET
24
20
THIRD PARTY CONTRACTOR
22
16
ON-LINE MERCHANT
18

Fig. 1

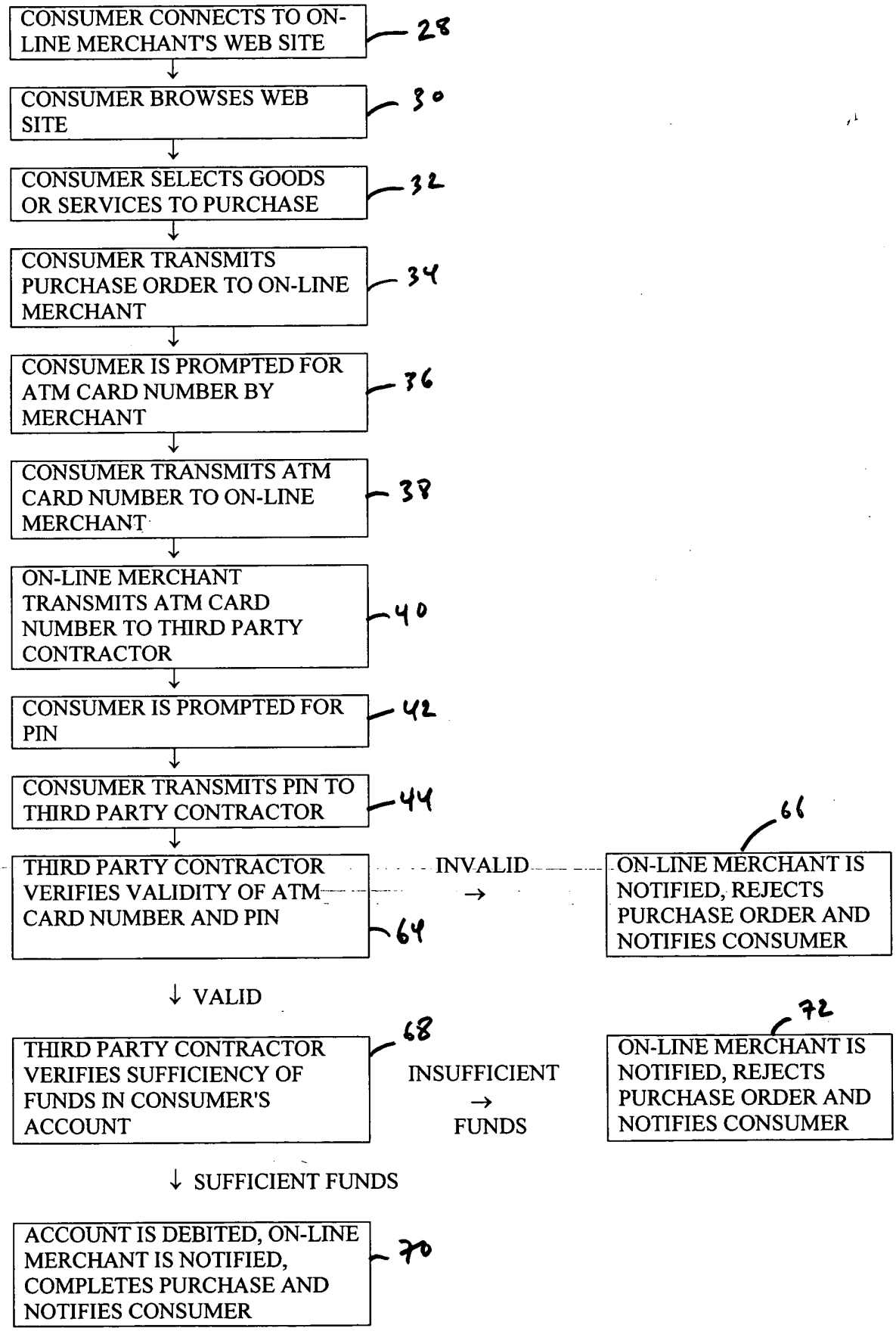
CONSUMER CONNECTS TO ON-LINE MERCHANT'S WEB SITE
28
CONSUMER BROWSES WEB SITE
30
CONSUMER SELECTS GOODS OR SERVICES TO PURCHASE
32
CONSUMER TRANSMITS PURCHASE ORDER TO ON-LINE MERCHANT
34
CONSUMER IS PROMPTED FOR ATM CARD NUMBER BY MERCHANT
36
CONSUMER TRANSMITS ATM CARD NUMBER TO ON-LINE MERCHANT
38
ON-LINE MERCHANT TRANSMITS ATM CARD NUMBER TO THIRD PARTY CONTRACTOR
40
CONSUMER IS PROMPTED FOR PIN
42
CONSUMER TRANSMITS PIN TO THIRD PARTY CONTRACTOR
44
THIRD PARTY CONTRACTOR VERIFIES VALIDITY OF ATM CARD NUMBER AND PIN
64
INVALID
→
ON-LINE MERCHANT IS NOTIFIED, REJECTS PURCHASE ORDER AND NOTIFIES CONSUMER
66
↓ VALID
THIRD PARTY CONTRACTOR VERIFIES SUFFICIENCY OF FUNDS IN CONSUMER'S ACCOUNT
68
INSUFFICIENT
→
FUNDS
ON-LINE MERCHANT IS NOTIFIED, REJECTS PURCHASE ORDER AND NOTIFIES CONSUMER
72
↓ SUFFICIENT FUNDS
ACCOUNT IS DEBITED, ON-LINE MERCHANT IS NOTIFIED, COMPLETES PURCHASE AND NOTIFIES CONSUMER
70

Fig. 2

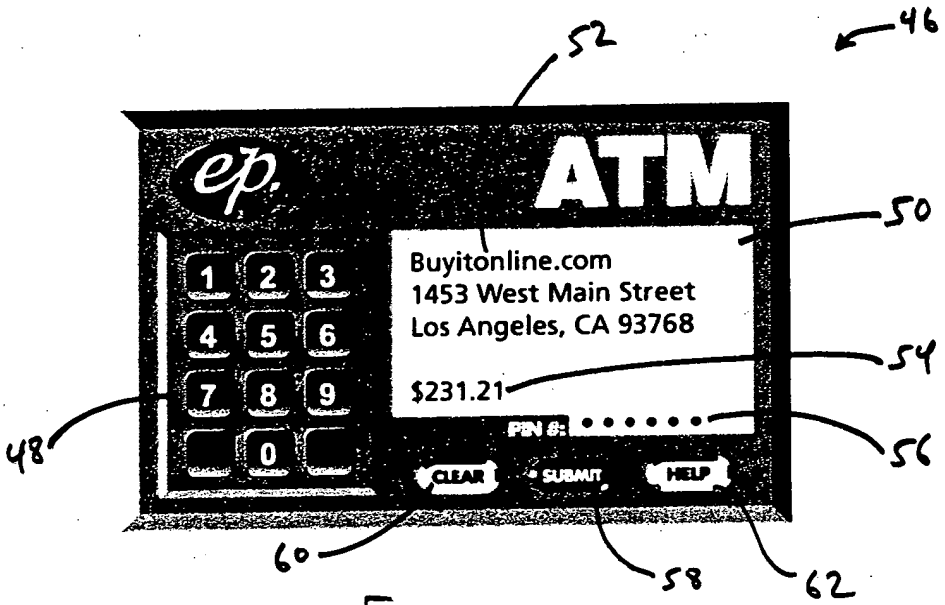
ATM
Buyitonline.com
1453 West Main Street
Los Angeles, CA 93768
$231.21
PIN #:
1 2 3
4 5 6
7 8 9
0
CLEAR
SUBMIT
HELP
46
52
50
54
48
56
60
58
62

Fig. 3

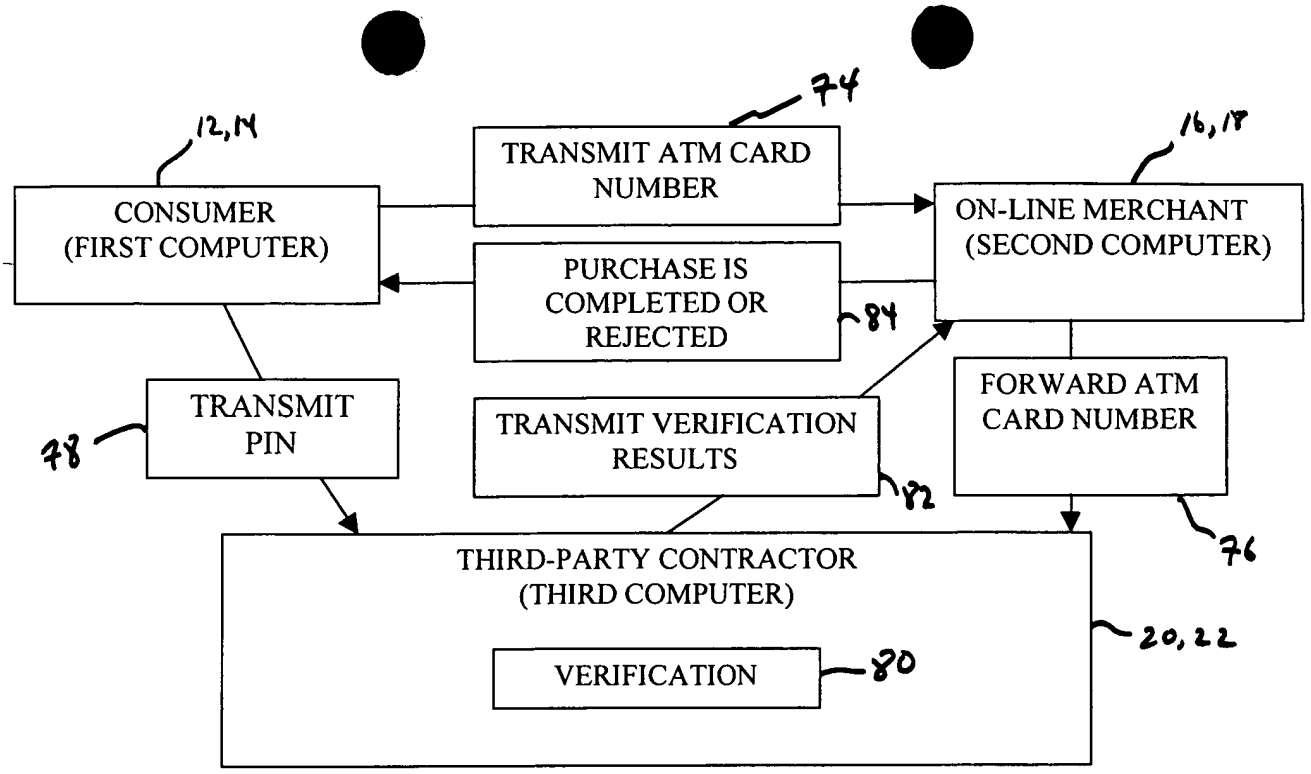

Fig. 4

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.util.Date ;
import com.ms.com.*;
import com.ms.asp.*;

public class JRoute
{
	public Socket socSocket;
	int m_iTimeout=10000;
	J8583 msg = new J8583();

	public int init(String input)
	{

		//VAR DECLARATIONS

		int port=0,ok=0;//CONNECTION PORT,CHECKSUM

		String hostname="localhost";//DEFAULT

		DataOutputStream theOutputStream;
		int parnum=8;
		String strInput="";

		String cardNumber="",amount="",expirydate="",trannum="",tid="",mid="",unique="",goAway="";

		try{

			////////////////////////////////READ INI PARS
			StringTokenizer tkToken = new StringTokenizer(input);

			hostname = tkToken.nextToken();
			port = Integer.parseInt(tkToken.nextToken());
			m_iTimeout = Integer.parseInt(tkToken.nextToken());



			////////////////////////////////

				//CARD NEEDS TO BE SENT TO OKTOPUS
				//BUILD MSG

				msg.addField(2,cardNumber);
				msg.addField(4,amount);
				msg.addField(14,expirydate);
				msg.addField(37,"1");
				msg.addField(41,tid);
				msg.addField(42,mid);
				msg.addField(61,unique);

				//CREATE SOCKET
				try
				{
				socSocket  = new Socket(hostname,port);
				socSocket.setSoTimeout(m_iTimeout);
				socSocket.setTcpNoDelay(true);
				}
				catch (UnknownHostException e)
				{
					return(-4); //HOST NOT FOUND
				}

				catch(IOException sockErr)
				{
					return(-3);
```

Fig. 5(a)

```
                                        }
                                        catch(Exception all)
                                        {
                                                return(-2);
                                        }

                                        msg.sendData(socSocket);


                        }
                        catch(Exception er)
                        {
                                return(-1); //SEND ERROR
                        }

                        return(-1);
                }

                public int listenfordata()
                {
                        //8583 CLASS
                        msg.receive(socSocket);
                        try
                        {
                                if(msg.decide(socSocket)==0) //APPROVAL
                                {
                                        try{
                                                return(0); //ITS GOOD
                                        }
                                        catch(Exception any)
                                        {
                                                return(-2); //ERROR
                                        }

                                }
                                else
                                {
                                        return(1); //DENIED
                                }
                        }
                        catch(Exception e)
                        {
                                return(-3); //ERRROR
                        }
                }
}

import java.io.*;
import java.net.*;

public class J8583
{
        private byte m_baOut[] = new byte[1024]; //OUTGOING BUFFER
        private int m_baOutIndex=0; //0 BASED INDEX OF FILLED BYTES
        private DataOutputStream m_dosData;
        private BufferedInputStream m_bisInput;
        private int m_field[] = new int[30];
        private     String m_value[] = new String[30];

        public J8583()
        {
                //CONSTRUCTOR
        }
        public void readFields()
        {
                int x=0;
                for(x=0;x<30;x++)
                        System.out.print(m_field[x]+"="+m_value[x]+"\n");
        }
```

Fig. 5(b)

```
public void addField(int field,String value)
{
	int x,j;
	j = value.length();
	m_baOut[m_baOutIndex] = (byte)field;
	m_baOutIndex++;

	for(x=0;x < j ;x++) //THE INDEX IS ONE HIGH TO LEAVE A NULL BETWEEN FIELDS
		m_baOut[x+m_baOutIndex] = (byte)value.charAt(x);

	m_baOutIndex += j+1;//RESET THE INDEX
}

public void sendData(Socket socLocal)
{
	try
	{
		//SEND
		m_dosData = new DataOutputStream(socLocal.getOutputStream());
		m_dosData.write(m_baOut,0,m_baOutIndex);
	}
	catch (UnknownHostException e)
	{
				System.out.print(e);
				System.exit(0);
	}
	catch(IOException sockErr)
	{
		System.out.print("Socket Connection: "+sockErr);
		System.exit(0);
	}
	catch(Exception all)
	{
		System.out.print("Socket Error: "+all);
		System.exit(0);
	}
}

public String resolveFieldValue(int fieldNumber)
{
	int x=0;
	for(x=0;x<30;x++)
		if(m_field[x]==fieldNumber)
			return(m_value[x]);
	return("");
}

public void receive(Socket socLocal)
{
try
	{
		m_bisInput = new BufferedInputStream(socLocal.getInputStream());

		int k=1,index=0;
		byte buf[] = new byte[1024];
		m_bisInput.read(buf,0,1024);

		for(k=0;k<30;k++)//INITIALIZE THE NULL STRINGS
			m_value[k]="";
		k=1;

		m_field[index] = buf[index];	//FIRST FIELD MARKED BY FIRST BYTE
		while(k<1024)
		{
			if(buf[k]!=0)
			{
				m_value[index]+=(char)buf[k++];
			}
			else
			{
```

Fig. 5(c)

```
                                        if(buf[k+1]==0)//END OF STREAM
                                                break;
                                        else
                                        {
                                                index++;
                                                m_field[index] = buf[k+1];
                                                //              System.out.print("|"+buf[k+1]+"|");
                                                k+=2;
                                        }
                                }

                        }



                }
                catch(IOException err)
                {
                        //TIMEOUT

                        //System.out.print((nTimeout)/1000+" Second Timeout");
                        try
                        {socLocal.close();}
                        catch(IOException Error){System.out.print("p"+Error);}


                }
                catch(Exception all)
                {
                        //MOST LIKELY A CLOSE ON IQ
                        System.out.print("Network Connection Closed " + all);
                        //redirect(urlTimeout);
                }
        }
        public int decide(Socket socLocal)
        {
                int k=0,index=0;
                byte pResult=0;

                for(k=0;k<30;k++)
                        if(m_field[k]==39)//GRAB PIN FIELD
                                pResult=(byte)m_value[k].charAt(0);

                        try{socLocal.close();}
                        catch(IOException e){}

                        if(pResult==48)//0 IS APPROVED
                        {
        //System.out.print("Thank You For Shopping At Electronic Paycheck");
                                return(0);
                        }
                        else
                        {
        //System.out.print("Denied");
                                return(1);
                        }
        }
}
```

Fig. 5(a)

```
// webhostDlg.cpp : implementation file
//

#include "stdafx.h"
#include "webhost.h"
#include "webhostDlg.h"

#include <afxtempl.h> // list

#ifndef TimeOut
#define TimeOut 200
#endif

#define TimerID 0x4000

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
        CAboutDlg();

// Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
        //}}AFX_VIRTUAL

// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
        //{{AFX_DATA_INIT(CAboutDlg)
        //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAboutDlg)
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
        //{{AFX_MSG_MAP(CAboutDlg)
                // No message handlers
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CWebhostDlg dialog

CWebhostDlg::CWebhostDlg(CWnd* pParent /*=NULL*/)
```

Fig. 6(a)

```
	: CDialog(CWebhostDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CWebhostDlg)
	m_in = 0;
	m_out = 0;
	m_q = _T("");
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CWebhostDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CWebhostDlg)
	DDX_Control(pDX, IDC_LST, m_lst);
	DDX_Text(pDX, IDC_IN, m_in);
	DDX_Text(pDX, IDC_OUT, m_out);
	DDX_Text(pDX, IDC_Q, m_q);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CWebhostDlg, CDialog)
	//{{AFX_MSG_MAP(CWebhostDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_WM_TIMER()
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

extern CWebhostApp theApp ;
CWebhostDlg* pDlg ;
char dbParam[256] ;

#include <ep_init.h>
#include <format.h>

#define __STDC__
#include <d3des.h>

EPsql sql ;
Listener listener ;
CList<Auth*,Auth*> Qa ;
CList<EndPoint*,EndPoint*> Qe ;
int matchF[]={ 2,14,41,42,61,0 } ; // f61=uniqueID, f44="5A315405018B44C4"
unsigned char key[]={ 0x29, 0xda, 0x91, 0x0b, 0x80, 0x9b, 0xfe, 0xd3 } ;

CString sDebug ;

void Listener::OnAccept(int nErrorCode) {
 EndPoint* tmp=new EndPoint() ;
 if (Accept(*tmp)) tmp->init() ; else delete tmp ;
}

int EndPoint::respond() {
 const char *p ;
 char pkt[1024],*s=pkt ;
 int i,d[]={ 35,43,47,48,52,62,102,103,0 } ;
 if (getType()==0) return 0 ;
 i=0 ; while (d[i]) { set(d[i],NULL) ; i++ ; }
 for (i=2; i<128; i++)
 { if (p=get(i)) { *s=i ; strcpy(s+1,p) ; s+=strlen(p)+2 ; } }
 return Send(pkt,s-pkt) ;
}

int EndPoint::aging(int t) {
 if (t) { if (t==-1) sec-- ; else sec=t ; }
 return sec ;
}
```

Fig. 6(5)

```
int EndPoint::match(M8583* m) {
 int f,i=0 ;
 while (f=matchF[i++]) if (strcmp(m->get(f),get(f))) return 0 ;
 return 1 ;
}

void EndPoint::init() {
 char buf[32] ;
 CString ip0 ;
 UINT port ;
 BOOL nodelay=TRUE ;
 SetSockOpt(TCP_NODELAY,&nodelay,sizeof(BOOL),IPPROTO_TCP) ;
 sec=TimeOut ; Qe.AddTail(this) ; pDlg->m_in++ ;
 GetPeerName(ip0,port) ; ip=inet_addr(ip0) ;
 sprintf(buf,"Connect %08x",ip) ; pDlg->note(buf) ;
}

void EndPoint::reject(int code) {
 char buf[32] ;
 sprintf(buf,"Reject %08x, code=%d",ip,code) ; pDlg->note(buf) ; sec=0 ;
 set(39,"100") ; set(44,buf+16) ; respond() ;
}

void EndPoint::OnReceive(int nErrorCode) {
 Auth* a ;
 EndPoint* e=NULL ;
 POSITION pos1,pos2 ;
 BOOL fullTrans=TRUE ;
 short len,l,i,f ;
 const char* pp ;
 char *p,pin[24],pan[20],buf[1024],scode[]="1200", offset[]="0000" ;
 if (nErrorCode) { sec=0 ; return ; }
 len=Receive(buf,1020) ; buf[len]=0 ; p=buf ; setType(1200) ; *pin=1 ;
 while (*p) {
  l=strlen(p) ; if ((*p==61)&&(l==2)&&(p[1]=='A')) fullTrans=FALSE ;
  if (set(*p,p+1,8)<1) { reject(*p) ; return ; }
  if ((*p==52)&&(l<14)) // clear PIN
  { *pin=0 ; pin[1]=l-1 ; strcpy(pin+2,p+1) ; memset(pin+l+1,15,10) ; }
  p+=(l+1) ; // build PIN block
 }
 if ((pp=get(52))&&(strncmp(pp,"F01",3)==0)) { reject(52) ; return ; }
 i=0 ; while (f=matchF[i++]) if (get(f)==NULL) { reject(f) ; return ; }
 if (*pin==0) { // got clear PIN, build PAN block, update PIN block
  strcpy(pan,offset) ; strncpy(pan+4,get(2)+strlen(get(2))-13,12) ;
  p=pin ; for (i=0; i<16; i++) { *p=(*p^pan[i])&15 ; p++ ; }
  for (i=0; i<8; i++) pin[i]=(pin[i*2]<<4)+pin[i*2+1] ;
  deskey(key,0) ; des((unsigned char*)pin,(unsigned char*)pan) ;
  for (i=0; i<8; i++) bin2hex(pin+i*2,pan[i]) ; pin[16]=0 ; set(52,pin) ;
  strcpy(buf,get(2)) ; strcat(buf,"=") ; strcat(buf,get(14)) ;
  strcat(buf,scode) ; strcat(buf,offset) ; set(35,buf) ;
 } // service code and offset hardcoded
 if (fullTrans) {
  pos2=Qe.GetHeadPosition() ;
  while (pos2) {
   pos1=pos2 ; e=Qe.GetNext(pos2) ;
   if (!match(e)||(e==this)) e=NULL ; else { Qe.RemoveAt(pos1) ; break ; }
  }
 }
 if (!fullTrans||fullTrans&&e)
 { a=new Auth(this,e) ; Qa.AddTail(a) ; Qe.RemoveAt(Qe.Find(this)) ; }
 sprintf(buf,"Recv %08x %d, card=%s",ip,len,get(2)) ; pDlg->note(buf) ;
}

Auth::Auth(EndPoint* e1, EndPoint* e2) {
 int i ;
 const char* p ;
 char f[16],dest[4]="N?" ;
 e[0]=e1 ; e[1]=e2 ; cp(*e1) ;
 if (e2) {
```

Fig. 6(c)

```
  if (e2->getType()==1200) setType(1200) ; set(3,"000000") ;
  for (i=2; i<128; i++) if (p=e2->get(i)) set(i,p) ;
 } else { set(3,"300000") ; set(4,"000000000000") ; }
 if (fillMsg(*this,sql,dbParam,3)) // 1: BIN, 2: mid/tid
 { e1->reject(1) ; if (e2) e2->reject(1) ; setType(0) ; return ; }
 id=++pDlg->m_out ; pDlg->UpdateData(FALSE) ;
 set(37,itoa(id,f,10),8) ; pDlg->m_ep.cp(*this) ;
 dest[1]=*(get(47)+1) ; pDlg->m_ep.send(dest) ;
}

BOOL Auth::isActive() {
 if ((e[1]==NULL)||(e[0]->aging(0)>0)&&(e[1]->aging(0)>0)) return TRUE ;
 setType(0) ; return FALSE ;
}

Auth::~Auth() {
 for (int i=0; i<2; i++) if (e[i])
 { e[i]->cp(*this) ; e[i]->respond() ; delete e[i] ; }
 setType(0) ;
}

/////////////////////////////////////////////////////////////////////////////
// CWebhostDlg message handlers

BOOL CWebhostDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	// Add "About..." menu item to system menu.

	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}

	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	// TODO: Add extra initialization here
 char IP[256],name[4],title[16] ;
 short TCPort,port ;
 const char fmt[]="%s %hd %2s %s %hd %s" ;
 const char usage[]="Usage: webhost IP port name DBparam listenPort" ;
 if (sscanf(theApp.m_lpCmdLine,fmt,IP,&TCPort,name,dbParam,&port)<5)
 { ::MessageBox(NULL,usage,"Error",MB_OK) ; EndDialog(0) ; return FALSE ; }
 sprintf(title,"WebHost %s %d",name,port) ; SetWindowText(title) ;
 if (!listener.Create(port)) {
  ::MessageBox(NULL,"Unable to create TCP/IP sockets.","Error",MB_OK) ;
  EndDialog(0) ; return FALSE ;
 }
 if (!listener.Listen()) {
  ::MessageBox(NULL,"Network error.","Error",MB_OK) ;
  EndDialog(0) ; return FALSE ;
 }
 if (m_ep.connect(IP,TCPort,name)) {
  ::MessageBox(NULL,"Error connecting to EProute.","Error",MB_OK) ;
  EndDialog(0) ; return FALSE ;
```

Fig. 6(d)

```
 }
 pDlg=this ; SetTimer(TimerID,1000,NULL) ;
        return TRUE;  // return TRUE  unless you set the focus to a control
}

void CWebhostDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CWebhostDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CWebhostDlg::OnQueryDragIcon()
{
        return (HCURSOR) m_hIcon;
}

void CWebhostDlg::OnTimer(UINT nIDEvent)
{
        // TODO: Add your message handler code here and/or call default
 Auth* a ;
 EndPoint* e ;
 POSITION pos1,pos2 ;
 BOOL del=FALSE ;
 if (nIDEvent==TimerID) {
  pos2=Qe.GetHeadPosition() ;
  while (pos2) {
   pos1=pos2 ; e=Qe.GetNext(pos2) ;
   if (e->aging()<1)  { Qe.RemoveAt(pos1) ; del=TRUE ; delete e ; }
  }
  pos2=Qa.GetHeadPosition() ;
  while (pos2) {
   pos1=pos2 ; a=Qa.GetNext(pos2) ;
```

Fig. 6(e)

```
    if (!a->isActive()) { Qa.RemoveAt(pos1) ; del=TRUE ; delete a ; }
   }
   if (del) note(NULL) ;
  }
        CDialog::OnTimer(nIDEvent);
}

void CWebhostDlg::note(const char* s) {
 if (s)
 { m_lst.AddString(s) ; if (m_lst.GetCount()>14) m_lst.DeleteString(0) ; }
 m_q.Format("%d %d",Qe.GetCount(),Qa.GetCount()) ; UpdateData(FALSE) ;
}

void On8583(short mType, EPacket* ep) {
 int n,i ;
 char s[64] ;
 const char* p ;
 POSITION pos1,pos2 ;
 Auth* a ;
 if ((mType==-1)||ep->mustExit()) { pDlg->EndDialog(0) ; return ; }
 if (mType) return ; // ignore other administrative messages
 ep->receive() ; if (ep->getType()==1430) return ;
 if (p=ep->get(37)) n=atoi(p) ; else return ;
 pos2=Qa.GetHeadPosition() ;
 while (pos2) {
  pos1=pos2 ; a=Qa.GetNext(pos2) ;
  if (a->match(n)) {
   ep->set(37,NULL) ; for (i=2; i<128; i++) if (p=ep->get(i)) a->set(i,p) ;
   Qa.RemoveAt(pos1) ; delete a ; pDlg->note(NULL) ; return ;
  }
 }
 pDlg->note("reversal") ; ep->getType(s) ; memset(s+4,'0',18) ; s[22]=0 ;
 if (p=ep->get(11)) strncpy(s+4,p,6) ;
 if (p=ep->get(12)) strncpy(s+10,p,12) ;
 if (p=ep->get(32)) strcpy(s+22,p) ;
 ep->set(56,s) ; ep->setType(1420) ; ep->send(ep->getSender()) ;
}
/*
sDebug.Format("") ;
::MessageBox(NULL,sDebug,"Debug",MB_OK) ;
*/
```

Fig. 6(f)